

ATTORNEY DOCKET No.

NVIDP015A/P001241

5

U.S. PATENT APPLICATION

FOR

Z-TEXTURE MAPPING SYSTEM, METHOD

AND COMPUTER PROGRAM PRODUCT

10

15

ASSIGNEE: **NVIDIA** CORPORATION

20

SILICON VALLEY IP GROUP, P.C.

P.O. Box 721120

SAN JOSE, CA 95172

Z-TEXTURE MAPPING SYSTEM, METHOD AND COMPUTER PROGRAM PRODUCT

5

RELATED APPLICATION(S)

The present application is a continuation of an application filed 10/02/00 under application serial number 09/678,111, which is incorporated herein by reference in its entirety for all purposes.

10

FIELD OF THE INVENTION

The present invention relates to computer graphics, and more particularly to mapping of depth values during computer graphics processing.

15

BACKGROUND OF THE INVENTION

20 Generally, bump and texture mapping are processes where basic contours of an object are expressed as a polygon in a modeling process and real world map data is used for a rendering process. During the course of bump and texture mapping, a color calculation is performed to incorporate colors onto an object in display coordinate space. This object with the colors is then displayed on a display device.

25 Prior Art Figure 1 illustrates the method by which an exemplary bump mapping process is accomplished. As shown, a primitive, i.e. polygon, triangle, etc., is first received with pixel data, as shown in operation 100. Included with such pixel data are normal values and possibly other values associated with the vertices associated with the polygon. These vectors are perspective and correctly

interpolated across the primitive. At each pixel, texture coordinates (also interpolated) are used to look up bump mapping information.

During bump mapping, the aforementioned normal values are modified
5 based on a bump map algorithm using the bump mapping information, as indicated
in operation 102 of Figure 1. In particular, the normal's direction is perturbed as
though the surface has been displaced a small amount in the direction of the
interpolated normals of the primitive. Figure 2 illustrates a primitive 200 with a
normal 202 that is modified to generate a perturbed normal 204. A bumpy surface is
10 thereby simulated.

Thereafter, lighting operations such as shading or the like are performed on
the pixel data using the perturbed normal values instead of the original normal
values, as indicated in operation 104. This method gives the appearance of bumps
15 and depressions in the surface. Also at this time, the color calculation may be
carried out in order to enhance the color of the pixel.

While the foregoing bump and texture mapping techniques feature the
unevenness of a surface and enhance the color of a pixel, they do not work well to
20 reflect any unevenness in shadows cast by or onto the bumpy surface. Further, there
are also limitations as to the interaction of geometric objects. These drawbacks are
mainly due to the fact that conventional bump and texture mapping processes have
no impact on the z-value of the pixel.

25 There is thus a need for a texture/bump mapping scheme during graphic
processing that overcomes these drawbacks for providing a more realistic rendered
image.

SUMMARY

A system, method and computer program product are provided for computer
5 graphics processing. In use, a value is modified based on an algorithm. An operation is
subsequently performed on pixel data taking into account the modified value.

BRIEF DESCRIPTION OF THE DRAWINGS

5 The foregoing and other aspects and advantages are better understood from
the following detailed description of a preferred embodiment of the invention with
reference to the drawings, in which:

Prior Art Figure 1 illustrates a bump mapping method of a prior art computer
graphics processing system;
10

Figure 2 illustrates a primitive with a normal that is modified to generate a
perturbed normal in accordance with a prior art bump mapping method;

Figure 3 is a schematic diagram showing an exemplary graphics subsystem
15 implementing bump mapping in tangent space according to the present invention;
and

Figure 4 is a flowchart illustrating a method for modifying depth-values in
addition to the normal values during bump mapping in accordance with one
20 embodiment of the present invention.

25

DETAILED DESCRIPTION

Bump and texture mapping are techniques to add more realism to synthetic images. Texture mapping adds realism by attaching images to geometric surfaces.

5 Bump mapping adds per-pixel surface relief shading, increasing the apparent complexity of the surface. Surfaces that should have associated fine grain details, small-scale geometric features, roughness, etc. are good candidates for bump mapping.

10 A bump map is an array of scalar or vector values that represents an object's features on a small scale. A custom renderer is used to map these height values into changes in the local surface normal. These perturbed normals are combined with the surface normal, and the results are used as inputs to a lighting equation at each pixel. In addition to using perturbed normals in such a manner, the present invention
15 further modifies depth-values to enhance graphics processing. It should be noted that one embodiment of the present invention may optionally modify z-values using a related, but separate map of scalar displacements, similar to traditional bump maps.

20 Figure 3 is a schematic diagram showing an exemplary graphics subsystem 300 implementing bump mapping in tangent space according to the present invention. A lighting and coloring module 310 includes a tangent space transform (TST) module 330 and a bump mapping module 340 which operate in a manner that is well known to those of ordinary skill. The graphics subsystem 300 further
25 includes a depth-value correction module 322. The manner in which the depth-value correction module 322 operates will be set forth hereinafter in greater detail. Also included is a memory 320 that stores output produced by the bump mapping module 340 in addition to output produced by the depth-value correction module 322.

30 The bump mapping module 340 and memory 320 work together to store bump maps by storing a normal vector in a texture map. Conventional RGB values

describe a normal vector relative to a coordinate frame defined at the vertices of the primitive, i.e. triangle. Three vectors including tangent, normal, and binormal vectors are interpolated, and the vector read from the texture map is then rotated by taking its dot product with each of the interpolated vectors, in turn. The result is the bump mapped normal which may be then processed by the lighting and coloring module 310 in a conventional manner.

Figure 4 is a flowchart 400 illustrating a method for computer graphics processing using the depth-value correction module 322 of Figure 3. First, in operation 401, pixel data is received including a depth-value. It should be noted that the depth value may include, but is not limited to a z-value, w-value, and/or any other value indicative of depth at least in part.

Thereafter, the depth-value is modified based on a depth-component of an algorithm. See operation 402. Such algorithm may include a bump map algorithm, texturing algorithm, etc. It should be noted, however, that any other desired algorithm may be utilized.

An operation is subsequently performed on the pixel data taking into account the modified depth-value, as indicated in operation 404. In one embodiment of the present invention, the operation may include a lighting operation. It should be noted, however, that the operation may take any form such as a hidden surface (z-test) calculation, shadow map operations, etc. A hidden surface calculation may determine visibility of various objects or portions thereof on the display device. Shadow map operations determine visibility with respect to another viewpoint such as a light source, thus permitting shadowing. During use, the foregoing operations function in a conventional manner, with the exception of using a modified depth-value as opposed to the original depth-value.

The technique for modifying or correcting the depth-values can be derived by perturbing eye-space value p_e using Equation #1.

Equation #1

5

$$p'_e = p_e + \Delta n_e$$

Perturbed eye-space value p'_e may then be run through a conventional projection transform, T_{proj} , associated with a viewing transformation that transforms the depth values from eye space to clip space. Clip-space z_c and w_c are thus extracted.

10

Thereafter, z_c and w_c are modified rendering z'_c and w'_c which are defined by Equations #2.

15

Equations #2

$$z'_c = z_c + \Delta (n \cdot T_{proj}[3])$$

$$w'_c = w_c + \Delta (n \cdot T_{proj}[4])$$

To perform per pixel calculation, z_c and $n \cdot T_{proj}[3]$ are iterated, and the value of Δ is read from a texture map. In some cases, bump mapping may be used in conjunction with displacement mapping. The displacement mapping may occur at one level of detail and filtering. Since the z-texture contains total displacements, there may be a mechanism to take this partial (filtered) displacement into account.

In such a situation, the vertices of the triangle may have already sampled the bump map once, and that earlier displacement may be subtracted from Δ in Equations #2. The result is set forth in Equations #3.

25

Equations #3

30

$$z'_c = z_c + \Delta_B (n \cdot T_{proj}[3]) - \Delta_D (n \cdot T_{proj}[3])$$

$$w'_c = w_c + \Delta_B (n \cdot T_{proj}[4]) - \Delta_D (n \cdot T_{proj}[3])$$

5 The values Δ_D are per vertex displacements already applied. The values Δ_B are values read in from the z-texture map.

10 It should be noted that the final depth value used and stored in the frame buffer may be computed by taking z_c/w_c with some appropriate scale and bias to place it in window coordinates. Further information regarding this chain of transformations may be found in the OpenGL[®] specification. Further, it should be understood that modifying the depth value may allow the lighting operation to display the interaction of displayed objects. Further, the modified depth value may allow the lighting operation to display bumpy shadows when applied to a typical shadow algorithm.

15

The present embodiment thus permits the per-pixel adjustment of the depth value of a polygon. The depth value of a polygon normally varies linearly, i.e. the polygon is planar. The present embodiment represents a mechanism by which the depth value is adjusted using a map, and the amount of adjustment is proportional/correct based on pre-projection coordinates. In effect, the modification has the projection transformation applied to it. It should be noted that this technique may be applied in contexts beyond bump mapping.

20

25 While various embodiments have been described above, it should be understood that they have been presented by way of example only, and not limitation. Thus, the breadth and scope of a preferred embodiment should not be limited by any of the above described exemplary embodiments, but should be defined only in accordance with the following claims and their equivalents.